**REMARKS**

Examiner rejects claims 1-36 under 35 U.S.C. 102(e) as being anticipated by Grun (US 6,629,166). Applicant's claim 1 includes the elements, "receiving from a processor a plurality of write transactions write combined in a processor," and "storing data associated with the plurality of write transactions in a buffer of an input/output (I/O) hub to form write combined data." Here, the data is stored in a buffer to form **write combined data**. Examiner stats that Grun discloses that requests from both of the message and data services are being combined in I/O controller 24. Grun's Figure 3 does illustrate and describe that MDS 30 "provide messaging and data transfer service to I/O controller 24." However, applicant's claim 1 does not include an element explicity directed towards combination of messages with data in a buffer, but rather combination of a plurality of write transactions, and specifically a combination of multiple data elements in the buffer of an I/O hub to form "write combined data." As the embodiment in applicant's description illustrates, three write transactions (24a-c) are received from a processor and data associated with write transactions 24a-c are buffered.

In contrast, Grun does not disclose or suggest combining multiple transactions or multiple elements in a buffer of an I/O hub to be later flushed to an I/O device. In fact, Grun illustrates use of buffers in Figure 6, not to combine a plurality of data associated with a plurality of write transactions, but rather to construct a message in the buffer and then pass a pointer and control of the buffer to a target service interface primitive (see col. 11 lines 27-34).

Furthermore, Grun does not contemplate write combining or a determination thereof in a processor. For claim 29, examiner points to col. 13 lines 26-56, which only discusses marking I/O controllers as ready, not ready, or failed in UMS 28. As can be seen, there is no discussion of write combining in a processor, determining write transaction to combine based on an attribute field, or

write combining in general. Additionally, UMS 28 is illustrated in TCA 22 that Examiner attempts to equate to applicant's I/O hub, not applicant's use of processor.

Similarly, claim 12 includes the element, "storage logic coupled to the receiving logic to store data associated with the first and second write transactions as write combined data." As mentioned above, Grun does not disclose "write combined data," but rather normal data transfer between devices, as illustrated in Figure 5. A single request, such as an I/O request, is sent from an initiating client, data transfer occurs, and an I/O completion is sent. In contrast, applicant's claim 12 includes a first and a second write transaction. The data is combined, i.e. storage logic is to store data associated with the first and the second to form write combined data, and the write combined data is flushed. Furthermore, in dependent claims 14 and 17, a first and a second completion signal are sent to the processor for the first and the second combined write transactions.

Applicant's claim 21 also includes, "to store data ...in the buffer to form a write combined data set." As aforementioned, Grun does not disclose, teach, or suggest write combining, but rather normal data transfer through channel based switch fabric, instead of a described bus-based computer system. Furthermore, Grun does not describe flushing "the write combined data set to the I/O device in response to a **protocol event.**" Examiner points to col. 7 lines 50-55 for a protocol event, such as a flushing signal. However, Grun only describes a request for I/O services from the initiator, an I/O controller that fulfills the request, and a completion from the I/O controller to the initiator. In contrast, applicant's claim 21 includes receiving a plurality of write transactions from the processor, storage of data in the buffer, a protocol event associated with the processor, and flushing to the I/O device. As can be seen, Grun does not describe any second or additional protocol event, such as a flush signal or latency condition, from or associated with the processor, but rather only a request, servicing the request, and a completion signal from the I/O controller back to the

initiator.

Applicant's claim 31 includes the element, "determining whether a latency condition exists, the latency condition including a delay in receiving a next combinable write transaction." Grun only discloses the following:

> Under ordered response protocol, after the target channel adapter has received a failure status for an operation, it will accept no further inbound target service interface primitives from the I/O controller until it has received a resync.indicate primitive for the service on which the error occurred. This primitive may signal to the target channel adapter that the I/O controller has recognized the failed operation and understands that any operation subsequent to it have been lost. After sending the resync primitive, the I/O controller may not be required to await an acknowledgment from the target channel adapter. The I/O controller may immediately begin posting target service interface primitives to the target channel adapter. An I/O controller may choose one of four actions after sending the resink primitive: it may choose to resume operations with the failed operation simply by reposting the failed target service interface primitive; it may choose to skip the failed operation and resume operations at a different point at the protocol flow; it may choose to send a message to the initiator indicating the loss of one or more messages, where this option gives a host the opportunity to
>
> attempt to reset an I/O controller and restart the associated service connections; or it may issue an unbind primitive,

As can be seen, Grun **requires receipt** of a failure status for an operation. However, as applicant described in an embodiment of a latency condition at paragraph 0024, a latency condition, for example, includes, "the lack of incoming write combinable writes." Furthermore, claim 31 includes, "the latency condition including a delay in receiving a next combinable write transaction," not receiving a failure status for an operation, as described in Grun.

Applicant's clam 34 includes both "storing ... to form a write combined data set," and "flushing...in response to a protocol event." As described above, Grun does not disclose combining or storing of data to form combined data and flushing that data in response to a protocol event, such as a flush signal or latency condition.

Therefore, Applicants respectfully submit that claims 1, 12, 21, 31, and 34 are in condition

for allowance, and furthermore, that dependent claims 2-11, 13-20, 22-30, 32-33, and 35-36 are also in condition for allowance for at least the same reasons stated above.

If there are any additional charges, please charge Deposit Account No. 50-0221. If a another telephone interview would in any way expedite the prosecution of the present application , the Examiner is invited to contact David P. McAbee at (503) 712-4988.

Respectfully submitted,
Intel Corporation

Dated: April 30, 2007

/David P. McAbee/Reg. No. 58,104/
David P. McAbee
Reg. No. 58,104

Intel Corporation
M/S JF3-147
2111 NE 25th Avenue
Hillsboro, OR 97124
Tele – 503-712-4988
Fax – 503-264-1729